XOR-Based Low-Cost Reconfigurable PUFs for IoT Security

XOR-Based Low-Cost Reconfigurable PUFs for IoT Security

WEIQIANG LIU, LEI ZHANG, and ZHENGRAN ZHANG, Nanjing University of Aeronautics and Astronautics, China
CHONGYAN GU, Queen's University Belfast, UK
CHENGHUA WANG, Nanjing University of Aeronautics and Astronautics, China
MAIRE O'NEILL, Queen's University Belfast, UK
FABRIZIO LOMBARDI, Northeastern University, USA

With the rapid development of the Internet of Things (IoT), security has attracted considerable interest. Conventional security solutions that have been proposed for the Internet based on classical cryptography cannot be applied to IoT nodes due as they are typically resource-constrained. A physical unclonable function (PUF) is a hardware-based security primitive and can be used to generate a key online or uniquely identify an integrated circuit (IC) by extracting its internal random differences using so-called challenge-response pairs (CRPs). It is regarded as a promising low-cost solution for IoT security. A logic reconfigurable PUF (RPUF) is highly efficient in terms of hardware cost. This paper first presents a new classification for RPUFs, namely circuit-based RPUF (C-RPUF) and algorithm-based RPUF (A-RPUF); two XOR-based RPUF circuits (an XOR-based reconfigurable bistable ring PUF (XRBR PUF) and an XOR-based reconfigurable ring oscillator PUF (XRRO PUF)) are proposed. Both the XRBR and XRRO PUFs are implemented on Xilinx Spartan-6 FPGAs. The implementation results are compared with previous PUF designs and show good uniqueness and reliability. Compared to conventional PUF designs, the most significant advantage of the proposed designs is that they are highly efficient in terms of hardware cost. Moreover, the XRRO PUF is the most efficient design when compared with previous RPUFs. Also, both the proposed XRRO and XRBR PUFs require only 12.5% of the hardware resources of previous bitstable ring PUFs and reconfigurable RO PUFs, respectively, to generate a 1-bit response. This confirms that the proposed XRBR and XRRO PUFs are very efficient designs with good uniqueness and reliability.

CCS Concepts: • **Security and privacy → Security in hardware**; **Embedded systems security**; • **Hardware →** *Reconfigurable logic applications*;

Additional Key Words and Phrases: Internet of Things (IoT), reconfigurable PUF, XOR, low cost

ACM Reference Format:
Weiqiang Liu, Lei Zhang, Zhengran Zhang, Chongyan Gu, Chenghua Wang, Maire O'Neill, and Fabrizio Lombardi. 2018. XOR-Based Low-Cost Reconfigurable PUFs for IoT Security. *ACM Trans. Embedd. Comput. Syst.* 1, 1 (August 2018), 21 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

This work was partially supported by the National Natural Science Foundation China (61771239), Nature Science Foundation of Jiangsu Province (BK20151477) and Fundamental Research Funds for the Central Universities China (NS2017024).
Authors' addresses: Weiqiang Liu; Lei Zhang; Zhengran Zhang, Nanjing University of Aeronautics and Astronautics, 29 Jiangjun Avenue, Nanjing, Jiangsu, 211106, China, {liuweiqiang,zhanglei_1993,zhengranzhang}@nuaa.edu.cn; Chongyan Gu, Queen's University Belfast, Belfast, UK, cgu01@qub.ac.uk; Chenghua Wang, Nanjing University of Aeronautics and Astronautics, Nanjing, China, chwang@nuaa.edu.cn; Maire O'Neill, Queen's University Belfast, Belfast, UK, m.oneill@ecit.qub.ac.uk; Fabrizio Lombardi, Northeastern University, Boston, USA, lombardi@ece.neu.edu.

1539-9087/2018/8-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

With the development of the Internet of Things (IoT), billions of nodes are now connected. Intelligent sensors and devices have been deployed, and the large amount of data generated by these devices necessitates the use of smart and autonomous machine to machine (M2M) communications. However, this also poses serious security and privacy issues which are a significant challenge for the IoT industry. The large number of devices opens up new attack vectors for criminal hackers to exploit, such as the IoT-based distributed denial-of-service (DDoS) attack on Dyn, which brought down Twitter, SoundCloud, Spotify, Reddit and a host of other sites [14]. Therefore, it is clear that IoT security is extremely important, thus attracing more and more research efforts in recent years [24]. Conventionally, the Internet is protected by classic cryptography, which is assumed to secure on well protected devices. However, most IoT devices are physically unprotected and can be easily captured by hostile attackers [22]. Also, it is very challenging to ensure the security of IoT devices on resource-constrained hardware platforms [3], such as sensors and radio frequency identification (RFID).

In recent years, new IoT security solutions have been proposed that consider new security primitives that originate from the hardware level [24]. Compared with software security, hardware security can increase the cost and difficulty of attacks, such as reverse engineering. At the same time, higher performance of encryption algorithms requires greater consumption of hardware resources. Moreover, classic encryption algorithms perform complex mathematical operations with keys. Therefore, the security of the cryptographic key determines the security of the entire system. Conventionally, the key is stored in non-volatile memory (NVM), which can be revealed by non-invasive or invasive attacks [12]. Furthermore, a classic cryptographic computation requires a substantial amount of resources and time, which are usually not available in IoT devices. Hardware-based security primitives can provide faster and safer authentication schemes.

The limited hardware resources of IoT nodes prompted the development of physical unclonable function (PUF). A PUF can extract the unique value from ICs from process variations that occur during the manufacturing process and quantify internal mismatches using binary sequences. Even if the structure of any two chips are exactly the same, there will be minor differences due to manufacturing variations. A PUF can extract these differences and provide each chip with a unique identification, alike to a digital fingerprint for the chip. As this IC fingerprint is obtained from the random difference in the circuit manufacturing process, even the chip manufacturer cannot duplicate a chip. In addition, the response of a PUF is electrical and automatically generated, and disappears when the power is turned off. A PUF can be used for on-line key generation or authentication [6]. For example, the key is generated from the PUF module only when needed. This is more secure than storing the key in NVM as used in the traditional encryption schemes.

PUF technology is considered to be particularly appropriate for IoT security due to its unique features, that is unclonable, low in cost and offers fast authentication. PUF can provide integrated and lightweight security primitives for secure communication in IoT [1]. Although the cost of PUF structures should be as low as possible for IoT devices, many PUF proposals in the literature suffer from consuming significant hardware resources. Hence, new lightweight PUF architectures are needed. In this paper, two XOR-based reconfigurable PUF circuits (namely, an XOR-based reconfigurable bistable ring PUF (XRBR PUF) and an XOR based reconfigurable ring oscillator PUF (XRRO PUF)) are proposed as cost-efficient logic reconfigurable PUF (RPUF) designs. Both designs are implemented on Field-Programmable Gate Arrays (FPGAs) and compared with previous RPUF designs. This paper is an extension of a previous work [27]; The main additional contributions are as follows:

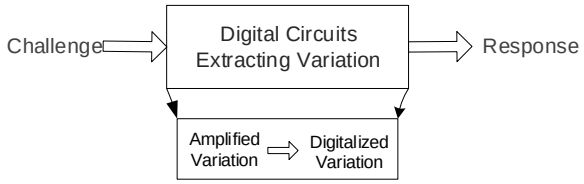


Fig. 1. Principle of the PUF circut.

- A new classification of RPUFs is proposed, circuit-based RPUF (C-RPUF) and algorithm-based RPUF (A-RPUF) is proposed.
- A novel XOR-based reconfigurable bistable ring PUF (XRBR PUF), is proposed.
- The hardware implementation of the XRRO PUFs on FPGA and detailed analysis are provided and compared with previous RPUFs. The results show that the proposed designs have improved the hardware efficiency significantly compared with previous RPUFs.

The rest of this paper is organized as follows: Section 2 provides background on PUF principles, metrics and typical structures. Section 3 summarizes the types of RPUF and proposes a new classification method. Section 4 presents the two proposed RPUF designs based on XOR gates. Section 5 provides the implementation of the proposed designs on Xilinx FPGAs. Section 6 shows the analysis of the simulation results. An efficiency analysis of hardware resource requirement is also provided in Section 6. Section 7 concludes the paper.

2 BACKGROUND

A PUF utilizes random differences in the IC manufacturing process to provide unique identity tags for chips. It can take in a challenge and then return a response as its output signal, as shown in Figure 1. The values of the challenge-response pairs (CRPs) [8] are unique for each chip. The CRPs are the manifestation of the random differences within the chip, which have the following characteristics:

- For the same chip, the responses $\{R_1, R_2, \ldots, R_n\}$ resulting from different challenges $\{C_1, C_2, \ldots, C_n\}$ should be mutually exclusive and unpredictable.
- For different chips, the responses $\{R_1, R_2, \ldots, R_n\}$ to the same challenges $\{C_1, C_2, \ldots, C_n\}$ should be mutually exclusive and unpredictable.

CRPs are used for authentication applications based on PUFs. The relationship between challenge and response is determined by the circuit itself and should be unique and ideally unpredictable. Therefore, a PUF is like a random process P with an input C that produces an output R, as defined in Equation (1). The characteristics of the random process are determined when the PUF is established, and the characteristics of each PUF are different.

$$PUF : C \rightarrow R : P(C) = R \tag{1}$$

The definition of $\mathcal{H}_{inter}$ and $\mathcal{H}_{intra}$ is introduced in Equation (2) and Equation (3) for use in the performance evaluation of the PUF.

$$\mathcal{H}_{inter} = HD\{P(C_1), P(C_2)\} \tag{2}$$

$$\mathcal{H}_{intra} = HD\{P_1(C), P_2(C)\} \tag{3}$$

$\mathcal{H}_{inter}$ is the inter-Hamming distance, and represents the Hamming distance of the responses obtained under the same challenge multiple times. The intra-Hamming distance shows the difference between the responses of several random processes $\{P_1, P_2, \ldots, P_n\}$ under the same challenge C.

2.1 PUF Metrics

To evaluate the performance of a PUF circuit, several metrics have been proposed, such as uniqueness and reliability [19]. These two metrics are the most commonly used to evaluate PUFs and are also used in this work. Furthermore, we propose a further metric, unit response cost (URC), to evaluate the cost efficiency of a PUF implemented on FPGA, as introduced in Section 6.

2.1.1 Uniqueness: Uniqueness measures the difference between the responses of a PUF, implemented on different chips, under the same challenge. The intra-Hamming distance, i.e. uniqueness, is given as follows:

$$U = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(P_i(C), P_j(C))}{n} \times 100\% \tag{4}$$

where k is the total number of chips, $P_i(C)$ is the response of the i^{th} chip and $P_j(C)$ is the response of the j^{th} chip under the same challenge C. The ideal value of uniqueness should be close to 50%, indicating the maximum difference in the responses of two PUFs. Uniqueness denotes the randomness of the responses, and therefore, it is also related to the security of a PUF.

2.1.2 Reliability: The reliability of a PUF measures the Hamming distance of the same PUF under various environmental conditions. In testing the reliability of a PUF, fluctuating temperatures are usually considered. The responses of the PUF chip are measured under different temperature gradients. Ideally, the response of a PUF should remain the same with the same challenge. However, as PUFs extract small differences in chips, the output of a PUF is unavoidably sensitive to a change in the operating environment. Reliability reflects the PUF's stability and its ideal value should be 100%, indicating that the PUF response value is stable and no bit flips occur in the response sequence under different test conditions. The equation for reliability is as follows:

$$R = \left(1 - \frac{1}{m} \sum_{t=1}^{m} \frac{HD(P(C_{i,t0}), P(C_{i,t}))}{n}\right) \times 100\% \tag{5}$$

where n is the sequence length of the response, m is the number of test. $P(C_{i,t})$ denotes the t-th sample of $P(C_i)$ among m repeated responses. For a PUF device, the reliability is established as 1 minus the average value of the inter-Hamming distance of the response samples under different operating conditions.

2.2 Traditional PUF Structures

The first PUF, or physical one-way function, proposed [21], was an optical PUF. It uses a laser to illuminate a particular dielectric at different incident angles θ to produce different interference patterns. It also quantifies the speckle pattern as a response to a one-way function that connects the excitation θ to a speckle pattern. It was found that ICs can be identified by comparing transistor drain current [18]. Silicon PUFs [8] were proposed for integration into standard digital ICs and have received widespread attention as a promising solution for low cost hardware security applications. As PUF circuits contain unique information on the underlying hardware variations, PUFs provide a promising approach to perform authentication [6] and IP-protection [10].

To date, a significant body of work has been reported on different PUF structures targeting both ASIC and FPGA based implementations; they can be categorized as memory-based and delay-based PUFs [23]. A SRAM PUF [10] is a type of memory-based PUF that is based on the uncertain transient characteristics of an SRAM cell (Figure 2): The SRAM is usually embedded in the chip and as such

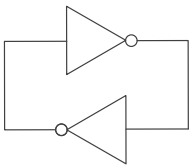


Fig. 2. Schematic diagram of a SRAM cell.

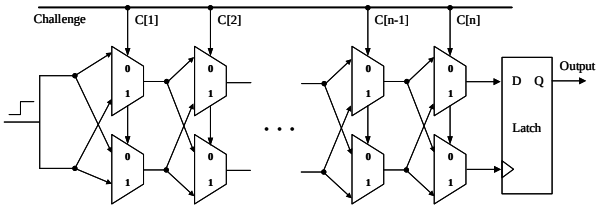


Fig. 3. Schematic diagram of Arbiter PUF [25].

consumes little additional hardware resources and is an inherent PUF. Guajardo et al. conducted extensive experiments on SRAM cells in FPGAs [11], which demonstrated that the uniqueness of the SRAM PUF was 49.97%, which is very close to the ideal value. The reliability interval was in the range of [88%, 96.43%]. However, an SRAM PUF cannot typically generate a sufficient number of CRPs. Hence, it is a known as a “weak" PUF [11] and is usually used as a key generator. There are also other memory-based PUF designs such as the butterfly PUF [15], and the TSRAM PUF [4].

Typical delay-based PUFs include the Arbiter PUF [25] and the Ring Oscillator (RO) PUF [6]. An arbiter-based PUF [25] was proposed by Lee et al. in 2004 (Figure 3); this PUF determines the output by comparing the signal propagation delay through two completely symmetrical signal transmission paths. Random deviations in manufacturing affect the delay path and cause random differences between the two paths. This leads to uncertain outputs, thus resulting in PUF behavior. However, it suffers from poor performance and is difficult to implement in FPGA. After testing, the response of the Arbiter PUF on 37 chips shows a uniqueness of 23% and a reliability of 95% [9]. The RO PUF (Figure 4) design exploits the difference in frequency between two identical ring oscillators, where each ring oscillator consists of an odd number of inverters (INVs). Due to random variations in the manufacturing process, even if each RO has the same structure, there will be a slight difference in the oscillation frequencies. The frequency measurement and comparison are accomplished by using rising edge counters and comparators. The process is as follows: whenever the rising edge of the periodic oscillation arrives, the count value of the edge counter is incremented by one; the counter counts over a specified period of time. In their period of time, the counter counts the number of rising edges of the two oscillating signals and the results of the counters are sent to the comparator. The result obtained from comparing the two count values is the PUF response. The uniqueness and reliability of an RO PUF are 46.15% and 99.52%, respectively, in [6]. A RO PUF is a “strong" PUF, as if it can provides a large number of CRPs. However, it also requires a significant amount of hardware resources.

A summary of the advantages and disadvantages of the three classic PUF architectures is given in Table 1. The SRAM PUF requires a small amount of hardware resource, but the number of CRPs is small. The Arbiter PUF and RO PUF can provide a sufficient number of CRPs. However, they require

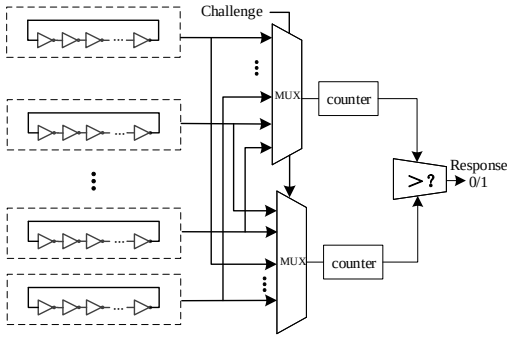


Fig. 4. Schematic diagram of a RO PUF [6].

Table 1. Advantages and Disadvantages of the Three Classic PUF Architectures

Types	Hardware Complexity	Number of CRPs	FPGA Implementation
SRAM PUF	Low	Small	Medium
Arbiter PUF	Medium	Large	Hard
RO PUF	High	Large	Easy

significantly more hardware resource than a SRAM PUF. The Arbiter PUF has further disadvantages in that it is applied to implement on single platform and it provides poor uniqueness. Finally, an RO PUF consumes a large amount of hardware resources making it difficult to implement on low-cost hardware platforms. Generally, memory-based and delay based PUFs have their drawbacks. Memory-based PUFs cannot provide many CRPs, while delay-based PUFs are not efficient in terms of hardware resource. Therefore, logic reconfigurable PUFs (RPUFs) have been studied [26] to address these problems and will be further discussed in the next section.

3 REVIEW AND CLASSIFICATION OF RECONFIGURABLE PUFS

Kursawe et al. were the first to propose a reconfigurable PUF (RPUF) circuit [16]. It increases the number of CRPs in adding some operations by reconfiguring the circuit. The reconfiguration signals can either partially or completely change the behavior of the PUF, thus resulting in a âĂIJnewâĂİ PUF. A RPUF can be classified as a physical RPUF or a logical RPUF [26]. The typical example of a physical RPUF is the reconfigurable optical PUF that uses a strong laser beam to quickly melt the optical medium, resulting in the rearrangement of optical scatters and so producing new random CRPs. This method completely changes the PUF from a physical molecular level; moreover, this change is irreversible. A logic RPUF generates a new PUF by configuring the excitation to reselect the signal transmission path or the logic gates. This method can increase the number of CRPs without changing the original structure of the PUF; therefore, the change is reversible.

This work is based on silicon-PUFs, and physical RPUFs are beyond the scope of this paper. Unless otherwise stated, the RPUFs mentioned in this work are all logic RPUFs. Next, RPUFs will be briefly reviewed using a new classification method that divides them into circuit-based RPUF (C-RPUF) and algorithm-based RPUF (A-RPUF).

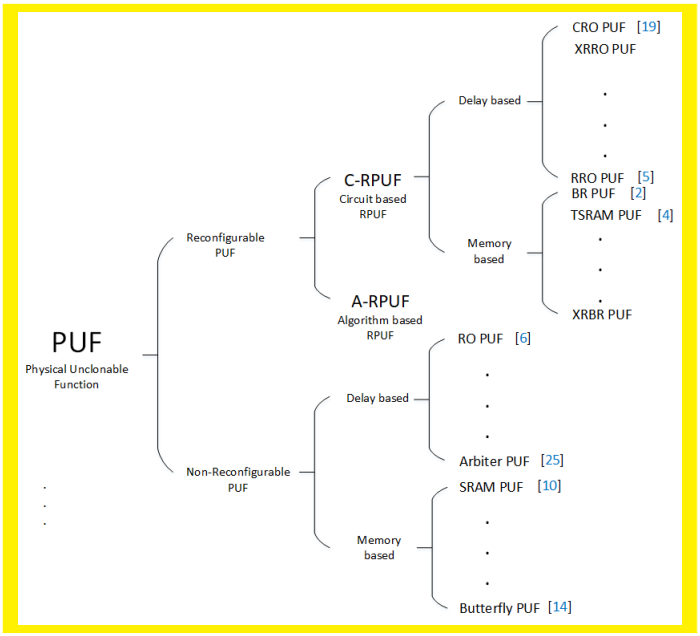


Fig. 5. The Taxonomy of PUF: reconfigurable and non-reconfigurable.

The new classification is based on the type of reconfiguration used in the design either circuit or at the algorithm level, as shown in Figure 5. CRO PUF, RRO PUF and BR PUF use a basic reconfigurable component to change the original structure without consuming significant hardware resources. However, an A-RPUF uses an algorithm and avoids the reconfigurable properties and thus avoids modification of the PUF structure.

3.1 Circuit based RPUF (C-RPUF)

A C-RPUF slightly increases the complexity of a circuit by adding logic gates to the original PUF and reconfiguration is performed with the logic gates. Several common C-RPUF structures will be reviewed next. A memory based PUF, namely the bistable ring-based PUF (BR PUF) was proposed in [2] and is shown in Figure 6(a). The basic principle of the BR-PUF is that an INV ring made of an even number of inverters has two possible stable states. Similar to a SRAM PUF, a ring with an even number of inverters falls into one of its two possible stable states when powered up or, more generally, when it is released from an unstable state. The BR PUF can be implemented on a Xilinx Virtex-II FPGA with the addition of multiplexers (MUXs) to make it a “strong" PUF, as shown in Figure 6(b). By applying different challenges, 2^n different rings can be generated. This implementation makes the BR PUF a reconfigurable PUF, but this also adds a cost: the increase in the number of INV leads to a remeasurement for the same number of MUXs. As shown in Figure 6(b), the BR-PUF needs n INVs and n MUXs to generate a 1-bit response.

Maiti and Schaumont proposed a configurable RO (CRO) PUF circuit (Figure 7) [20]. Each RO can be reconfigured by using a multiplexer to select one of two inverters that are connected to the multiplexer to form an RO. The selection nodes $\{S_1, S_2, \ldots, S_n\}$ assigned to the selection port of the multiplexers act also as configurable signals. This design has been implemented on Xilinx Spartan-3E FPGAs and experimental results have shown that the uniqueness and reliability are 47.31% and 99.14%, respectively.

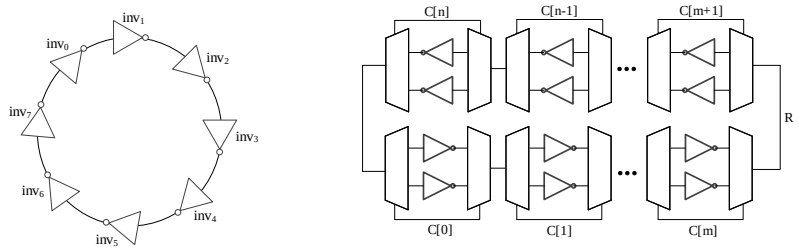


Fig. 6. The BR PUF: (a) the Schematic of a bistable ring, and (b) Structure [2].

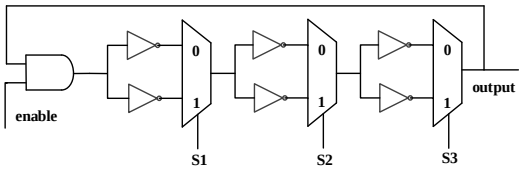


Fig. 7. The Structure of the CRO PUF proposed in [20].

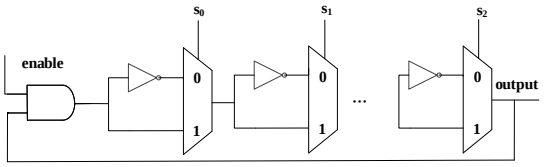


Fig. 8. The Structure of the CRO PUF proposed in [7].

Another CRO PUF was proposed in [7]; it replaces the RO of [6] with a delay unit. Each delay unit consists of a MUX and an INV (Figure 8). The selection signals determine the INV that contributes to the RO. As for the RRO PUF design proposed in [5] (Figure 9), the selection nodes $\{S_1, S_2, \ldots, S_n\}$ of the multiplexers that previously acted as configurable nodes, are now used to establish the transmission path for forming an RO. The difference originates from the fabrication variations in the multiplexer and the wires directly connected to the multiplexer. The RRO PUF has been implemented on Xilinx Spartan-6 FPGAs and the results show better performance in termes of uniqueness and reliability; these metrics are 49.97% and 98.41%, respectively.

The above three types of C-RPUF are configured by adding a MUX to the RO unit, thus changing the structure of the RO. Although this method requires slightly more hardware resources, it can produce more CRPs compared to the original RO PUF. Therefore, the CRO PUF [7], the PUF designed by Maiti and Schaumont 2011 and the RRO PUF [5] are more efficient in terms of hardware than a traditional RO PUF [6].

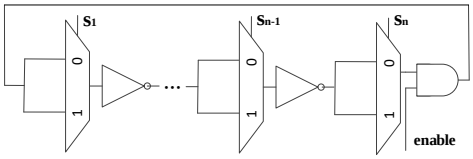


Fig. 9. The Structure of a RRO PUF [5].

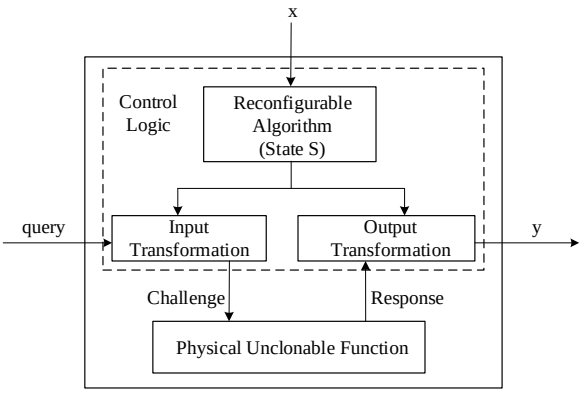


Fig. 10. The A-RPUF model proposed in [13].

3.2 Algorithm based RPUF (A-RPUF)

The principle of an A-RPUF is to combine the PUF circuit with a configurable algorithm to ensure a higher degree of diversity in the mapping between challenge and response. This not only increases the number of CRPs, but it also improves the security of the entire structure through the algorithm. [13] proposed an A-RPUF model (Figure 10). The reconfiguration algorithm is dependent on the input parameter x so that the control logic maintains a state S. The query signal is sent to the PUF circuit through the input transfer function. The responses of the PUF are returned to the output transfer function to produce the result y. Each query signal x and corresponding signal y pairing can be used as CRPs of the entire A-RPUF. Since the mapping of the input and output functions is controlled by the reconstruction algorithm, the A-RPUF structure can be updated by changing the value of x. The security of this A-RPUF model is also related to the algorithm.

4 PROPOSED XOR BASED RPUF

4.1 Properties of XOR Gates

This section introduces and discusses two novel RPUF circuits. XOR gates play an important role in the proposed RPUFs. Let the two inputs to an XOR gate be denoted as A and B, and the output be given as Q. As a function of A, the output is as follows: (1) When A="0", Q= B, the XOR gate acts as a buffer, so relaying the input value to the output following a delay. (2) When A="1", Q= $\overline{B}$ and the XOR gate operates as an inverter, i.e., the output value is the logic inverse of the input. Therefore, an XOR gate can be regarded as a configurable inverter gate by controlling one of its inputs. Thus, the logic function can be switched from a Buffer to an INV by controlling one of the

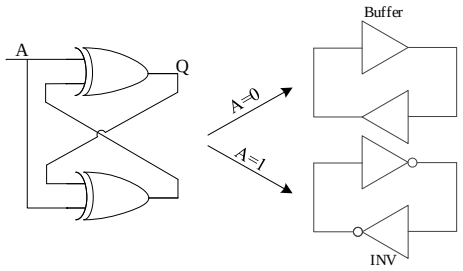


Fig. 11. The X-Latch circuit.

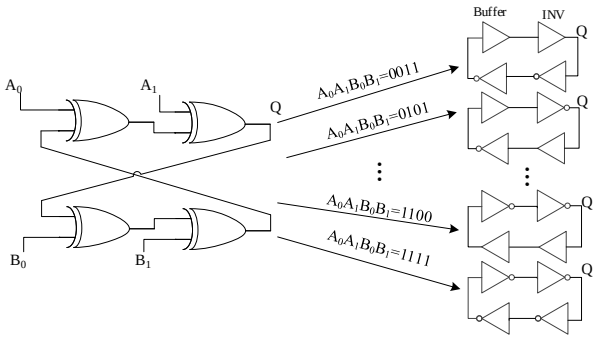


Fig. 12. Configuration mode of 2X-Latch circuit.

inputs of the XOR gate. Whereas the reconfiguration of the RPUF, discussed in Section 3, that is CRO PUFs and BR PUFs, is based on the basic MUX, the basic reconfigurable component of the proposed RPUF is an XOR gate.

4.2 The XOR-based Reconfigurable Bistable Ring PUF (XRBR PUF)

In a CRO PUF [20], the reconfigurable design is based on a MUX. The designer can add a MUX to the RO circuit and transform the original circuit into a configurable circuit. When the circuit is configured, a new circuit is generated. Since an XOR gate can be configured as a Buffer or INV, it can also be converted to a configurable gate. If INV is replaced by an XOR gate in an SRAM PUF and the extra inputs of the two XOR gates are connected together, the X-Latch (XOR gate based latch) circuit is generated. As shown in Figure 11, when the configuration signals of an X-Latch circuit are “0" and “1", respectively, the circuit consists of two mutually coupled buffers and two mutually coupled INVs. When A = 1, the X-Latch circuit can be reconfigured as a SRAM PUF circuit.

The X-Latch circuit can be expanded to a 2X-Latch by increasing the number of XORs, thus creating two 2-Latch coupled circuits (Figure 12). The 2X-Latch can be reconstructed into 8 different circuits by configuring the signals $\{A_0A_1B_0B_1\}$.

Consider the configuration signal $\{A_0A_1B_0B_1 = 0101\}$ as an example: The 4 XORs will be reconfigured as a Buffer, INV, Buffer, and INV respect for each configuration signal. A buffer unit can be considered as a delay wire that does not effect the logic function of a circuit. Hence, the

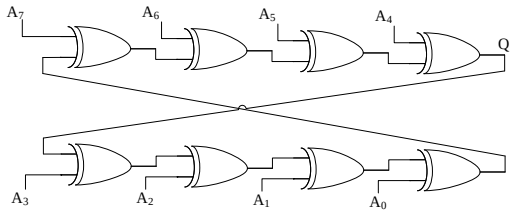


Fig. 13. The schematic diagram of the proposed XRBR PUF

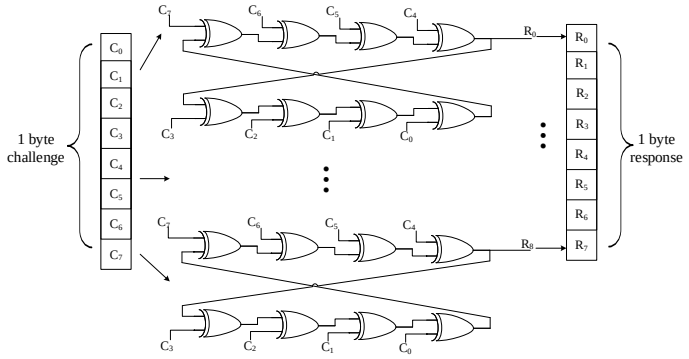


Fig. 14. The 1-byte response of XRBR PUF.

X-Latch circuit can be reconstructed as a SRAM by the configuration signal and equivalently, the 2X-Latch circuit can be reconfigured as 6 different SRAM PUF circuits by configuring different signals. An even number of logic levels "1" must be presented in the bit sequence of the configuration signal $\{A_0A_1B_0B_1\}$, thus the 4-bit configuration signal has a different configuration i.e., $2^4/2 = 8$. Note that for the configuration $\{A_0A_1B_0B_1 = 1111\}$ where all 4 XORs are reconstructed into INVs, any two connected INVs can be combined to form a buffer, equivalent to a SRAM PUF circuit.

Similar to an SRAM PUF, there are slight differences in characteristics such as delay and driving capability for the different XOR reconstructed Buffers and INVs in the presence of random process variations when manufacturing integrated circuits. Therefore, the electrical characteristics of the circuit constructed by different connections are not the same and under different configurations, the output of the circuit returns different responses, i.e., the output Q is different.

Furthermore, by utilizing appropriate configurations, 3X-Latch, 4X-Latch, ... nX-Latch can be configured to create different memory-based PUF circuits.

As the reconfigurable property of the X-Latch, an XOR gate based reconfigurable bistable ring PUF (XRBR PUF) is proposed as shown in Figure 13. The XRBR PUF consists of 8 XOR gates; one input of the 2-input XOR is used for the reconfiguration data. Each XRBR generates a 1-bit output response. In the case of a 1-byte response, an 8 XRBR PUF structure is needed (Figure 14).

4.3 The XOR-based Reconfigurable Ring Oscillator PUF (XRRO PUF)

A RO with a 5-stage inverter is shown in Figure 15; the oscillation frequency of the RO is affected by different parameters such as ambient temperature, noise and electromagnetic fields. Therefore,

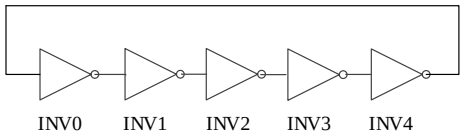


Fig. 15. Traditional 5-stage RO.

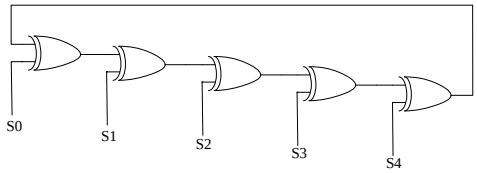


Fig. 16. 5-stage XOR gates based RO.

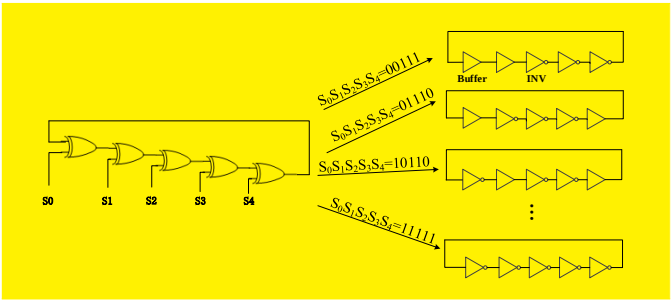
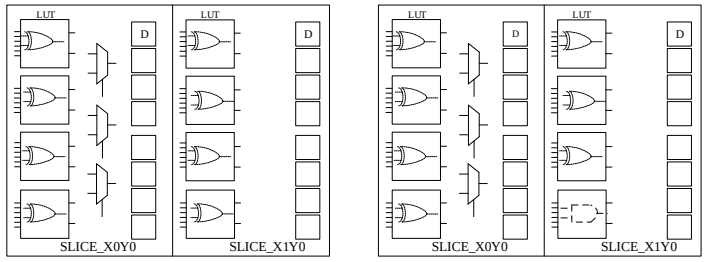


Fig. 17. Configuration method of the XOR gate based RO.

even if two ROs with exactly the same characteristics are utilized, the oscillation frequencies will be different. As discussed previously, an XOR gate can be used in place of an INV in the 5-stage RO (Figure 16).

As shown in Figure 16, one input node of the XOR gate is connected to the configuration signal $\{S_0S_1S_2S_3S_4\}$ and the other input node is connected to the output of the previous XOR. The number of INVs in the loop must be odd. Hence, the XOR loop in Figure 16 can be reconstructed using $2^5/2 = 16$ different ROs by setting the configuration signal $\{S_0S_1S_2S_3S_4\}$. This process for the configuration of the XOR gate-based RO is shown in Figure 17. Consider the configuration for $\{S_0S_1S_2S_3S_4 = 00111\}$ as an example. The 5 XORs will be reconfigured as: Buffer, Buffer, INV, INV and INV, respectively. A buffer unit can be considered as a delay wire that does not affect the logic function of the circuit. Hence, the entire circuit is constructed as a 3-stage RO using the configuration signal.

The XOR based ROs can construct up to 16 different ROs from different configurations. To transform this structure to a PUF design, measurements have been performed to show the differences between the oscillation frequencies of 16 ROs constructed by different configuration signals, which is further discussed in section 5.



(a) (b)

Fig. 18. The utilization of CLBs for (a) XRBR PUF (no enable signal) and (b) XRRO PUF (enable signal provided by the AND gate).

Table 2. Oscillation Frequency Results for the Four Configuration Signals.

Configuration Signal	Oscillation Frequency (MHz)
10101	157
01010	152
11100	155
11111	161

The next section gives a detailed presentation of the design of the proposed XRBR PUF and XRRO PUF. The XRBR PUF is feasible as a memory-based PUF and to generate a sufficient number of CRPs. It provides more outputs without requiring additional hardware resources. The efficiency of these two designs will be further analyzed in Section 6.

5 FPGA IMPLEMENTATIONS OF XRBR AND XRRO PUFS

FPGAs are usually used as a verification platform to assess the performance of PUFs. Xilinx Spartan-6 FPGAs are utilized to implement the proposed XRBR PUF and XRRO PUF. These FPGAs have an adequate number of CLBs; there are 8 LUTs, 3 MUXs and 16 D-flip-flops in a configurable logic block (CLB). In this paper, the XRBR PUF is generated using 8 XOR gates, which requires 8 LUTs of a CLB on the Xilinx Spartan-6 FPGA as shown in Figure 18(a). An XRRO PUF is generated using 7 XOR gates and one AND gate, which requires 8 LUTs in a CLB of the Xilinx Spartan-6 FPGA as shown in Figure 18(b).For the XRBR PUF, 8 XRBRs are needed to generate a 1-byte response. Thus, a 1-byte response has a cost of 8 CLBs for the Spartan-6 FPGA.

We also implemented a 5-stages XRRO PUF (Figure 16) on a Xilinx Spartan-6 XC6SLX9 FPGA and connected the output signal through the I/O port to the oscilloscope and measured an oscillation frequency of the signal. In this implementation, only the oscillation signal under four types of configuration modes was measured, as shown in Figure 19. It shows the oscillation waveforms for the configuration signals “10101", “01110", “11100" and “11111". The oscillation frequencies under these configuration signals are given in Table 2.

The FPGA implementation results show that there are differences in oscillation frequency for the ROs in the configurations as caused by process variations. Therefore, the architecture can be used as a PUF design. When the number of XORs in the XRRO PUF scheme is altered, the oscillation

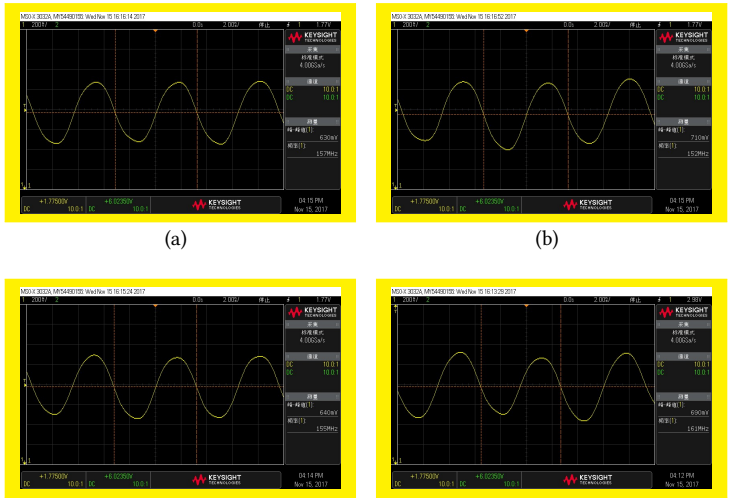


Fig. 19. RO waveforms for the configurable signals: (a) 10101 (Frequency: 157 MHz) (b) 01010 (Frequency: 152 MHz) (c) 11100 (Frequency: 155 MHz) and (d) 11111 (Frequency: 161 MHz).

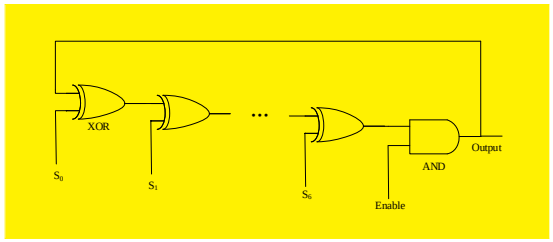


Fig. 20. The schematic diagram of the proposed XRRO PUF.

frequencies of the constructed ROs also change. Moreover, more XOR stages implemented in a circuit, a larger number of ROs can be constructed.

Therefore, the INV-based RO is replaced by an XOR-based RO in the RO PUF and an XRRO PUF circuit is proposed as a type of delay-based RPUF. Each XRRO consists of 7 XORs and 1 AND gate as shown in Figure 20. The configuration signal $\{S_0S_1S_2S_3S_4S_5S_6\}$ controls the construction of these XOR gates and determines the oscillation start time of the entire loop. The “Enable" signal is the activation signal, hence it is equivalent to a RO switch. When “Enable = 1", the AND gate behaves as a buffer and the behavior of the entire circuit is determined by the configuration of these XORs. When “Enable = 0", the output of the AND gate is always at a low logic level. Hence, the output state of the entire loop is controlled by the AND gate. The method of using the AND gate to control the operating state of the entire circuit yields a low-power design. As there are a large number of ROs in a RO PUF, the overall system would require a high power if all of ROs are constantly oscillating. However, in the proposed circuit not all ROs participate in generating the response. Therefore, by controlling the Enable signal, only the selected ROs participate in the oscillation response. All other unselected signals are in a non-oscillating state, which greatly reduces the overall power consumption.

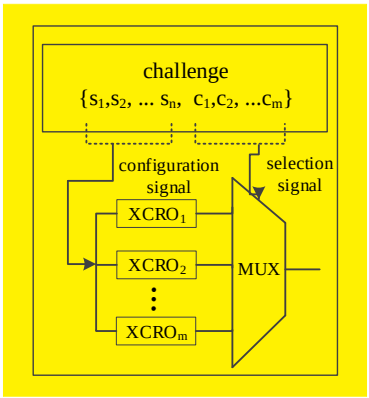


Fig. 21. The challenge setting mode of the XRRO PUF design.

Figure 21 describes the challenge setting mode of this design: one of the XOR gate inputs is used as a configuration signal. Therefore, the signal set $\{s_1, s_2, \ldots, s_n\}$ is also part of the l-bit PUF challenge. By applying different configuration signals, the XOR gates are selected and used in configuring the RO. Different ROs generate various responses under the same configuration signal. In this design, the configuration signal is taken from the n-bit at the left-hand side of the challenge. The remaining m-bits of the challenge are sent to the select signal of the MUX (where $l = n + m$). In this configuration method, the XRRO PUF can form many different ROs. This configuration mode provides more outputs without using additional hardware resources, because the challenge not only chooses what ROs for comparison, but it is also used to construct a new RO. More outputs can be obtained by changing the configuration signal in the stimulus to avoid increasing additional hardware resources.

The implementation of the XRRO PUF is more complicated compared with the XRBR PUF. The architecture of the XRRO PUF is given in Figure 22: The excitation signal selects the pair of ring oscillators that is involved in the generation of the response for the configuration signals $\{S_0S_1S_2 \ldots S_n\}$ to the XRROs. The output signal of each RO is connected to the trigger port of the rising edge counter. When the rising edge of the oscillation signal arrives, the counter value increases by 1. Then when the control module enables the RO pair, the ring oscillator starts to oscillate. The enabled RO pair starts a timer that is controlled by the system clock. When the count reaches a specified value, it sets the enable signal to “0" and the RO pairs stop the oscillation. At this time, the count value of the 2 counters “$cnt1$" and “$cnt2$" are sent to the comparator. If “$cnt1 > cnt2$" the frequency of XRRO1 is larger than XRRO2, then the response is “1"; otherwise, the response is “0".

In the XRRO-PUF, the control module is the top-level module and controls the timing operation of the XRRO arrays, counting the modules, and the timer modules. The counting module mainly completes the comparison of the count value. The timer is driven by the enable signal of the top-level module from the system clock.

In theory, the frequency of each RO should be independent of its physical position in the FPGA. However, [17] has shown that the RO frequency distribution shows a dependency on the layout of the Spartan FPGA. For example, the RO frequencies show a centrosymmetric distribution in the Xilinx Spartan-3E FPGA. So the frequencies of the ROs that are placed closer to the border, are always lower than those located in the center area (Figure 23).

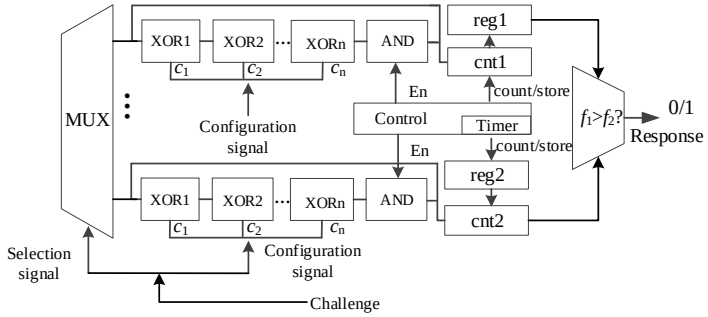


Fig. 22. The Architecture of the XRRO PUF.

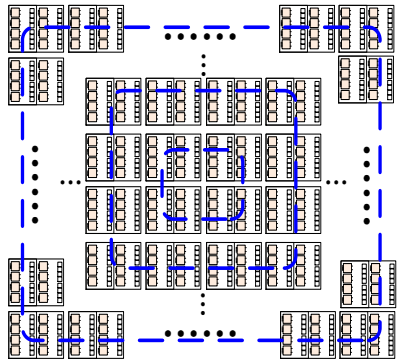
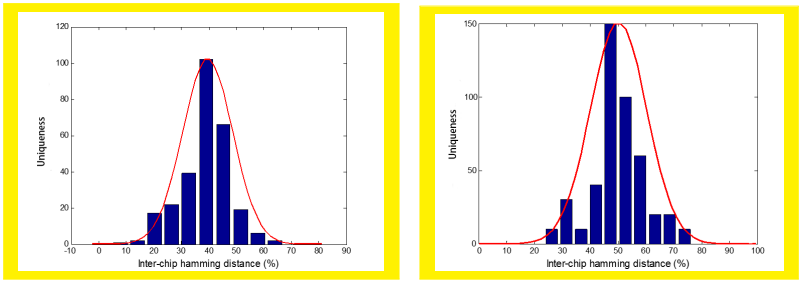


Fig. 23. Floorplan of XRRO PUF in FPGA.

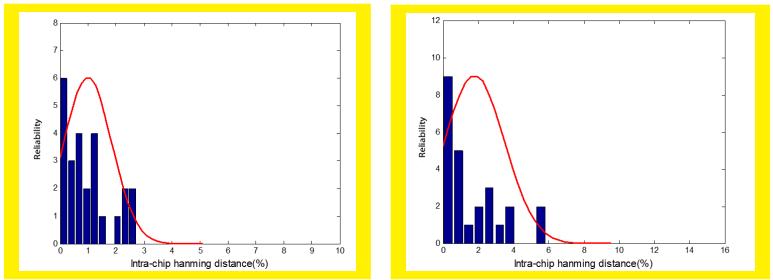
These frequency characteristics could make it possible for attackers to predict the response of a RO PUF. Hence, several comparison strategies have been proposed to design high security RO PUFs [17]. To avoid a frequency attack and poor uniqueness, the loop comparison strategy presented in [17] is selected for the XRRO PUF design. According to their frequency distribution, the ROs located in the same annular around the center have a similar frequency. In the process of response generation, the frequency comparison of XRROs within the same ring can reduce the prediction possibility and improve the security of the PUF. The loop comparison strategy is used considering both efficiency and security.

The implementation of the loop comparison strategy is shown in Figure 23; the selected ROs belongs to the same ring. 8 rows and 8 columns of CLBs are used to form 64 XRROs using the Spartan-6 FPGA. The frequencies of the XRROs are compared according to the location (indicated by the dashed loop lines in Figure 23) and hard macros are utilized to fix the location of each XCRO into a single CLB made of two slices.



(a) (b)

Fig. 24. Uniqueness of (a) XRBR PUF and (b) XRRO PUF.



(a) (b)

Fig. 25. Reliability of (a) XRBR PUF and (b) XRRO PUF.

6 COMPARISON AND ANALYSIS

6.1 Results Comparison for Proposed RPUFs

The proposed XRBR PUF and XRRO PUF designs are implemented on 8 Xilinx Spartan-6 XC6SLX9 FPGAs, at a working temperature range of 0°C-70°C. As mentioned in Section 2, the performance of a PUF is usually determined by two main parameters, i.e. uniqueness and reliability [19]. Uniqueness is used to evaluate the ability to differentiating two different devices, while reliability is used to measure the stability of a PUF at different temperatures and environments. Both uniqueness and reliability are analyzed and calculated by the Hamming distance of a PUFâĂŹs response. For the proposed XRBR PUF and XRRO PUF, these two metrics are analyzed and compared with previous PUFs implemented on FPGAs. The red lines in Figures 24- 25 show Gaussian distributions with the mean values from the test results as ideal values for uniqueness and reliability.

As shown in Figure 24, the uniqueness values for the proposed XRBR PUF and XRRO PUF are 40.67% and 48.76%, respectively. The reliability of the proposed XRBR PUF and XRRO PUF are shown in Figure 25. The values are 98.22% and 97.72% respectively, which are very close to the ideal value, i.e. 100%. The number of CRPs used to obtain the uniqueness and reliability results is 450. These uniqueness and reliability results are also compared with previous designs and the results are reported in Table 3 and Table 4. Table 3 shows that the XRBR PUF achieving the best reliability. Its uniqueness is also significantly improved compared with the BR PUF [2]. Although, it is not as good as a traditional SRAM PUF [10], the proposed XRBR PUF can generate CRPs and is also more cost efficient as further illustrated in the next section. Table 4 shows that the proposed

Table 3. Average Uniqueness and Reliability of Memory-based PUFs

PUF Designs	Uniqueness	Reliability
SRAM PUF [10]	49.97%	96.43%
BR PUF [2]	34.80%	98.04%
Proposed XRBR PUF	40.67%	98.22%

Table 4. Average Uniqueness and Reliability of Various RO PUFs

PUF Designs	Uniqueness	Reliability
RO PUF [6]	46.15%	99.52%
CRO PUF [20]	47.31%	99.14%
CRO PUF [7]	47.31%	95.95%
RRO PUF [5]	49.97%	97.40%
Proposed XRRO PUF	48.76%	97.72%

XRRO PUF has better uniqueness compared with the conventional RO PUF [6] and the CRO PUF [20], [7]. However, it is slightly worse compared with the RRO PUF [4]. The reliability of the RRO PUF is worse than for the proposed XRRO PUF. In summary, the proposed XRBR and XRRO PUFs show good performance in terms of uniqueness and reliability.

For a reconfigurable PUF design, its hardware utilization efficiency is an important feature to consider. The efficiency of the proposed XRBR PUF and XRRO PUF designs is analyzed in next section and compared with previous PUFs.

6.2 Cost Efficiency For PUF Designs Implemented on FPGAs

As PUFs are a promising security primitive for IoT devices, a low-cost PUF design is highly desireable. In this work, cost efficiency is defined for a PUF design implemented on FPGA. The CLB is the basic component of an FPGA and the number of CLBs required for generating a 1-bit response is defined as the Unit Response Cost (URC). URC is used as a metric to compare the cost efficiency of different PUF designs that are implemented on FPGAs. If a PUF design uses the same number of CLBs to generate more responses, then it is considered to be a more efficient design. Thus, the lower the URC, the higher the efficiency of the design. Based on this observation, the following metric is proposed and used to quantify the URC of a PUF design:

$$URC = \frac{N}{R_{bit}} \tag{6}$$

where R_{bit} denotes the number of response bits and N denotes the number of CLBs. So for a PUF design, a lower URC value implies a higher efficiency. Xilinx FPGAs are used in this work and a utilization of CLBs is shown in Figure 18.

As shown in Figure 6(a), a BR PUF needs 4 INVs and 4 MUXs (at least) to generate 4 possible SRAM cells using a 2-bit configuration of challenges. It is impossible to implement it in one CLB because there are no enough MUXs. Using 3 CLBs, which include 24 LUTs and 12 MUXs, the BR PUF can produce $2^6 = 64$ possible bits using a 6-bit configuration of challenges. Hence, on average, 1 CLB can produce 21 possible response bits, which means the URC of BR PUF is 1/21:

$$URC_{BR} = 1/16 \tag{7}$$

Similarly, the XRBR-PUF is fixed inside the FPGA's CLB. Unlike the RO PUF, the XRBR-PUF generates a 1-bit response by configuring the challenge without the need for comparisons between CLBs. Thus, the efficiency of the XRBR-PUF is also a fixed value. It does not change with n. Therefore, in the same CLB, the XRBR-PUF can reconstruct $2^7 = 128$ different BR cells by changing the configuration signal, hence corresponding to 128 possible responses. Compared with the BR PUF, this has been improved significantly. The URC of the XRBR-PUF is:

$$URC_{XRBR} = 1/128 = URC_{BR}/8 \tag{8}$$

In [6], a RO PUF generates the response bits by comparing the delay difference between a pair of ROs, in which every RO consists of a closed-loop chain of inverters (and the number of inverters must be odd). C_2^n bits of response can be generated using a RO array with n CLBs. The cost for conventional RO PUFs to generate a response bit is:

$$URC_{RO} = \frac{2n}{n(n-1)} \tag{9}$$

The CRO and RRO PUFs proposed in [20] and [5] can generate 8 different ROs in a CLB. Each comparison between two ROs located in different CLBs with the same configuration signals can generate a response bit. Thus, $8C_2^n$ distinct pairs of ROs are implemented in n CLBs. The number of response bits is $8n(n-1)/2$ and the URC of the CRO PUF and RRO PUF is:

$$URC_{CRO} = URC_{RRO} = \frac{2n}{8n(n-1)} \tag{10}$$

The proposed XRRO PUF can implement 64 different ROs in a CLB. Each comparison between two different ROs generates a bit. Therefore, $64C_2^n$ distinct pairs of ROs are generated in n CLBs. The number of response bits is $64n(n-1)/2$ and the URC of the XRRO PUF is:

$$URC_{XRRO} = \frac{2n}{64n(n-1)} \tag{11}$$

By considering Equations (9-11), the following expression is found for the cost:

$$URC_{RO} = 8 \times URC_{CRO} = 8 \times URC_{RRO} = 64 \times URC_{XRRO} \tag{12}$$

The URC results of the proposed designs are shown in Figure 26 and compared with previous RPUFs. These results illustrate that the efficiency of the XRBR PUF is the highest when $n < 5$; the efficiency of the XRRO PUF is the highest when $n > 5$; and the efficiencies of the XRRO PUF and CRO/RRO PUF are higher than XRBR-PUF when $n > 33$, which is summarized as follows:

$$\begin{cases} C_{XRBR} < C_{XRRO} < C_{BR} < C_{CRO/RRO} < C_{RO} & n < 5 \\ C_{XRRO} < C_{XRBR} < C_{CRO/RRO} < C_{BR} < C_{RO} & 5 < n < 33 \\ C_{XRRO} < C_{CRO/RRO} < C_{XRBR} < C_{RO} < C_{BR} & n > 33 \end{cases} \tag{13}$$

The typical URCs for RO, CRO, RRO, BR, XRBR and XRRO PUFs are listed in Table 5. In general, the XRBR PUF has the highest efficiency under limited resources and it only requires 12.5% of the hardware used by the BR PUFs to generate a 1-bit response. The XRRO PUF is the most efficient design in most cases and also utilizes only 12.5% of the hardware required by previous CRO and RRO PUFs to generate a 1-bit response.

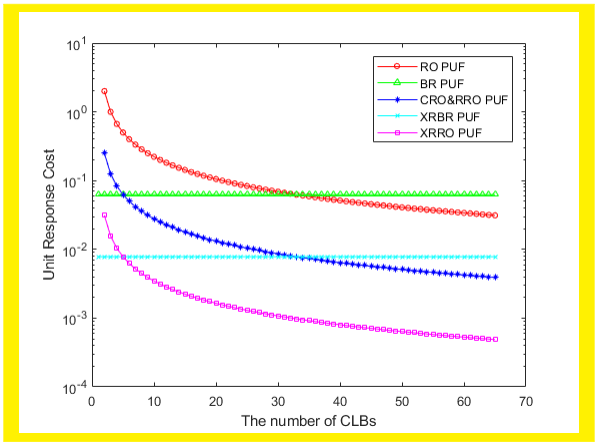


Fig. 26. URCs of RO, CRO, RRO, BR, XRBR and XRRO PUFs.

Table 5. Typical URCs of RO, CRO, RRO, BR, XRBR and XRRO PUFs

Types	n = 2	n = 4	n = 8	n = 16	n = 32	n = 64
RO PUF	2.0000000	0.6666667	0.2857142	0.1333333	0.0645161	0.0317460
CRO PUF	0.2500000	0.0833333	0.0357142	0.0166667	0.0080645	0.0039683
RRO PUF	0.2500000	0.0833333	0.0357142	0.0166667	0.0080645	0.0039683
BR PUF	0.0625000	0.0625000	0.0625000	0.0625000	0.0625000	0.0625000
XRBR PUF	0.0078125	0.0078125	0.0078125	0.0078125	0.0078125	0.0078125
XRRO PUF	0.0312500	0.0104167	0.0044643	0.0020833	0.0010081	0.0004960

7 CONCLUSION

This paper has proposed a new classification for RPUFs, namely C-RPUFs and A-RPUFs. Two XOR gate-based RPUF designs (XRBR PUF and XRRO PUF) have been proposed. The XRBR PUF used 8 XOR gates and allow memory-based PUF to produce sufficient CRPs to address the disadvantage of a conventional SRAM PUF. In the XRRO PUF, the XOR gates replace the inverters in a conventional RO PUF. By using one input of the XOR gate as a control signal, the XOR gates can be used as configurable inverters. Hence, more responses can be generated due to the larger number of configurable ROs with different configuration signals. The proposed two RPUF designs have been implemented using Xilinx Spartan-6 FPGAs and each PUF is designed using a CLB to improve its uniqueness. Both proposed PUFs show great uniqueness compared with previous designs. The XRRO PUF is the most efficient design. It only uses 12.5% of the hardware resources required by previous CRO and RRO PUFs to generate a 1-bit response. Moreover, the XRBR PUF also uses 12.5% of the hardware resources of BR PUFs to generate a 1-bit response. This confirms that the proposed XRBR PUFs and XRRO PUFs are very efficient designs with good uniqueness and reliability.

REFERENCES

[1] Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. 2017. A PUF-based secure communication protocol for IoT. *ACM Trans. Embed. Comput. Syst.* 16, 67 (April 2017), 1–25.

[2] Qingqing Chen, György Csaba, Paolo Lugli, Ulf Schlichtmann, and Ulrich Rührmair. 2011. The bistable ring PUF: a new architecture for strong physical unclonable functions. In *IEEE International Symposium on Hardware-Oriented*

Security and Trust (HOST). 134–141.
[3] Shanzhi Chen, Hui Xu, Dake Liu, and Bo Hu. 2014. A vision of IoT: applications, challenges, and opportunities with china perspective. *IEEE Internet of Things Journal* 1, 4 (2014), 349–359.
[4] Yijun Cui, Chenghua Wang, Weiqiang Liu, and Maire O'Neill. 2016. A reconfigurable memory PUF based on tristate inverter arrays. In *IEEE International Workshop on Signal Processing Systems (SiPS)*. 171–176.
[5] Yijun Cui, Chenghua Wang, Weiqiang Liu, Yifei Yu, Maire O'Neill, and Fabrizio Lombardi. 2016. Low-cost configurable ring oscillator PUF with improved uniqueness. In *IEEE International Symposium on Circuits and Systems (ISCAS)*. 558–561.
[6] Srinivas Devadas and G. Edward Suh. 2007. Physical unclonable functions for device authentication and secret key generation. In *Design Automation Conference (DAC)*. 9–14.
[7] Mingze Gao, Khai Lai, and Gang Qu. 2014. A highly flexible ring oscillator PUF. In *Design Automation Conference (DAC)*. 1–6.
[8] Blaise Gassend, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas. 2002. Silicon physical random functions. In *ACM Conference on Computer and Communications Security (CCS)*. 148–160.
[9] Blaise Gassend, Daihyun Lim, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas. 2004. Identification and authentication of integrated circuits. *Concurrency & Computation Practice & Experience (CCPE)* 16, 11 (2004), 1077–1098.
[10] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, and Pim Tuyls. 2007. FPGA intrinsic PUFs and their use for IP protection. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*. 63–80.
[11] Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, Pim Tuyls, Geert Jan, and Schrijen. 2013. Physical unclonable functions, FPGAs and public-key crypto for IP protection. *Journal of Radioanalytical Chemistry* 40, 1 (2013), 233–235.
[12] Marc Joye and Jean Jacques Quisquater. 2002. *Hessian Elliptic Curves and Side-Channel Attacks*. Springer Berlin Heidelberg. 335–345 pages.
[13] Stefan Katzenbeisser, unal Kocaba, Vincent Van Der Leest, Ahmad Reza Sadeghi, Geert Jan Schrijen, and Christian Wachsmann. 2011. Recyclable PUFs: Logically Reconfigurable PUFs. *Journal of Cryptographic Engineering* 1, 3 (Sep 2011), 177.
[14] KrebsonSecurity. [n. d.]. DDoS on Dyn Impacts Twitter, Spotify, Reddit. ([n. d.]). https://krebsonsecurity.com/2016/10/ddos-on-dyn-impacts-twitter-spotify-reddit/ Accessed: 08-11-2016.
[15] Sandeep S. Kumar, Jorge Guajardo, Roel Maes, Geert Jan Schrijen, and Pim Tuyls. 2008. The butterfly PUF protecting IP on every FPGA. *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)* (2008), 67–70.
[16] Klaus Kursawe, Ahmad Reza Sadeghi, Dries Schellekens, Boris Skoric, and Pim Tuyls. 2009. Reconfigurable physical unclonable functions - enabling technology for tamper-resistant storage. In *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*. 22–29.
[17] Weiqiang Liu, Yifei Yu, Chenghua Wang, Yijun Cui, and Maire O'Neill. 2015. RO PUF design in FPGAs with new comparison strategies. In *IEEE International Symposium on Circuits and Systems (ISCAS)*. 77–80.
[18] Keith Lofstrom, W. Robert Daasch, and Donald Taylor. 2002. IC identification circuit using device mismatch. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 372–373.
[19] Roel Maes. 2016. *Physically Unclonable Functions: Constructions, Properties and Applications*. Springer Publishing Company, Incorporated. 49–80 pages.
[20] Abhranil Maiti and Patrick Schaumont. 2011. *Improved Ring Oscillator PUF: An FPGA-friendly Secure Primitive*. Springer-Verlag New York, Inc. 375–397 pages.
[21] Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. 2002. Physical one-way functions. *Science* 297, 5589 (2002), 2026.
[22] Farzana Rahman and Sheikh Iqbal Ahamed. 2012. Looking for needles in a haystack: detecting counterfeits in large scale RFID systems using batch authentication protocol. In *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM)*. 811–816.
[23] Ahmad Reza Sadeghi and David Naccache. 2010. Towards hardware-intrinsic security: foundations and practice. *Information Security & Cryptography* 364, 1849 (2010), 3215–3230.
[24] John A. Stankovic. 2014. Research directions for the Internet of Things. *IEEE Internet of Things Journal* 1, 1 (2014), 3–9.
[25] Jae W. Lee, Daihym Lim, Blaise Gassend, and G. Edward Suh. 2004. A technique to build a secret key in integrated circuits for identification and authentication applications. In *Symposium on VLSI Circuits*. 176–179.
[26] Le Zhang, Zhi Hui Kong, Chip Hong Chang, and Alessandro Cabrini. 2014. Exploiting process variations and programming sensitivity of phase change memory for reconfigurable physical unclonable functions. *IEEE Transactions on Information Forensics & Security (TIFS)* 9, 6 (2014), 921–932.
[27] Lei Zhang, Chenghua Wang, Weiqiang Liu, Maire O'Neill, and Fabrizio Lombardi. 2017. XOR gate based low-cost configurable RO PUF. In *IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–4.